Отчет по реимплементации StyleGAN-NADA

Введение

Проведена исследовательская работа по изучению и реимплементации метода редактирования изображений **StyleGAN-NADA** (Рис.1). Целью работы является дообучение генеративной модели для создания изображений из выбранного домена, включая абстрактные или нереалистичные образы. Особенностью метода является то, что для обучения не требуются изображения целевого домена, а используется текстовое описание цели, что исключает необходимость в сборе и разметке дополнительных данных.

За основу был выбран **StyleGAN2**, архитектура которого состоит из Mapping Network (преобразует случайный шум в стилизующие латентные векторы W) и Synthesis Network (использует эти векторы W для генерации изображения) (Рис.2). Мы научились модифицировать Synthesis Network генератора, при этом сохраняя структуру и свойства Mapping Network и общего скрытого пространства. Такой подход после сравнительно быстрого обучения позволяет генерировать неограниченное число разнообразных изображений в выбранном домене.

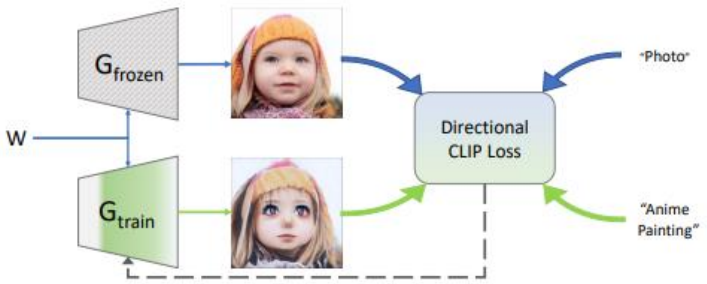

Рисунок 1. StyleGAN-NADA.

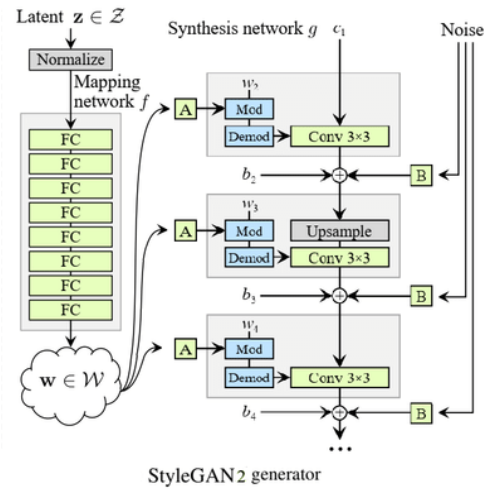

Рисунок 2. StyleGAN2.

Для формирования текстового представления (эмбеддинга) использовалась модель **CLIP (Contrastive Language-Image Pre-training)**. CLIP представляет собой мультимодальную модель, которая "понимает" связь между изображениями и текстом, сопоставляя их в единое семантическое пространство. CLIP способен оценивать, как должен измениться стиль изображения на основе текстового запроса, предоставляя векторные представления как для изображений, так и для текста, что позволяет измерять семантическую схожесть между ними (Рис 3).

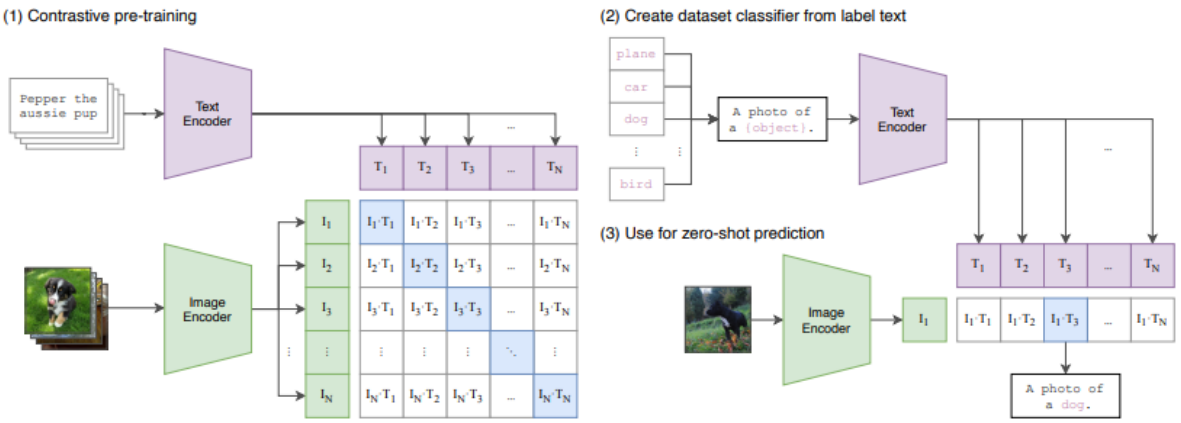

Рисунок 3. CLIP (Contrastive Language-Image Pre-training).

Принцип работы

Обучение производится с использованием двух экземпляров генератора, которые изначально являются идентичными копиями предобученного StyleGAN2.

1. Первый генератор (`model_frozen`) замораживается и используется для генерации изображений из исходного домена. Он служит "якорем" или точкой отсчета.
2. Второй генератор (`model_train`) обучается генерировать изображения целевого домена.

Основная идея обучения заключается в том, чтобы изменить "направление" генерации `model_train` таким образом, чтобы оно соответствовало "направлению" в пространстве CLIP, заданному текстовыми подсказками. Это означает, что `model_train` адаптируется, чтобы перемещать вектор изменения изображения вдоль направления вектора между исходным текстовым описанием (`source_text`) и целевым текстовым описанием (`target_text`) в пространстве CLIP.

Для этого используется направленная функция потерь **Directional CLIP Loss**. Эта потеря стремится выровнять:

- Вектор изменения в CLIP-пространстве изображений (полученный из `CLIP(generated_img_train) - CLIP(generated_img_frozen)`).
- С вектором изменения в CLIP-пространстве текста (полученный из `CLIP(target_text) - CLIP(source_text)`).

Направленная функция потерь использует косинусное расстояние между этими двумя векторами направления и определяется как:

$$L_{CLIP\_direction} = 1 - cosine\_similarity(CLIP(I_{train}) - CLIP(I_{frozen}), CLIP(T_{target}) - CLIP(T_{source})) \quad (1)$$

Где:

- I_{train} — изображение, сгенерированное обучаемым генератором.
- I_{frozen} — изображение, сгенерированное замороженным генератором.
- T_{target} — текстовая подсказка целевого домена.
- T_{source} — текстовая подсказка исходного домена.

Функция потерь (Loss Function)

Общая функция потерь (`Total Loss`) считается как сумма двух основных компонент:

$$L_{total} = \lambda_{CLIP} \cdot L_{CLIP\_direction} + \lambda_{L2} \cdot L_2 \quad (2)$$

1. $L_{CLIP\_direction}$ (CLIP Directional Loss): Это основная потеря, которая направляет адаптацию генератора в пространстве CLIP. Она заставляет сгенерированные изображения двигаться в направлении, указанном текстовыми подсказками.
2. L_2 (L2 Reconstruction Loss): Это потеря реконструкции, она измеряет потерю между сгенерированными изображениями I_{train} и I_{frozen}, помогает сохранить общую структуру и детали исходных изображений, не позволяя генератору слишком сильно отклоняться от реалистичности или терять идентичность при трансформации.

Динамическая настройка коэффициентов потерь

Для решения задачи по подбору коэффициентов λ_{CLIP} и λ_{L2} был применен динамический метод с использованием градиентного спуска и отдельного оптимизатора `optimizer_lambda`, который обновляет эти коэффициенты.

Преимущества такого подхода:

- Автоматическая настройка: Вместо трудоемкого ручного подбора весов потерь (`lambda_clip`, `lambda_l2`), модель автоматически находит их оптимальные значения на основе общей функции потерь.
- Динамическая адаптация: Веса потерь могут изменяться в процессе обучения, адаптируясь к текущему состоянию модели и градиентов. Например, если $L2$ становится слишком большим (модель сильно отклоняется от исходного пространства), а $L_{CLIP\_direction}$ уже достаточно мал, оптимизатор может автоматически увеличить λ_{L2} и/или уменьшить λ_{CLIP}, чтобы сфокусироваться на сохранении структуры. Это обеспечивает более сбалансированное обучение.
- Меньше ручной работы: Уменьшает необходимость в обширном поиске гиперпараметров, что значительно ускоряет экспериментальный цикл.

Выбор слоев для разморозки в обучаемом генераторе

Известно, что не все слои генератора одинаково важны для внесения стилистических или доменных изменений. Некоторые слои отвечают за низкоуровневые признаки (общая структура), другие — за высокоуровневые детали (текстуры, цвет, "стиль").

Для поиска наиболее чувствительных к изменениям слоев, которые будут разморожены (то есть, будут участвовать в обучении), опробованы различные методы:

1. Разморозка только поздних слоев Synthesis Network (Эксперимент 1, `freeze_layers`):

Подход: В качестве эксперимента были разморожены только поздние слои синтезирующей сети (Synthesis Network). Эти слои, как правило, отвечают за генерацию более мелких деталей, текстур и "стиля" в высоком разрешении. Размораживание происходило один раз, перед циклом обучения.

Результат: Результат обучения получился неудовлетворительным, что, вероятно, указывает на недостаточное влияние этих слоев для правильной адаптации к новым, более глобальным доменам (Рис.4).

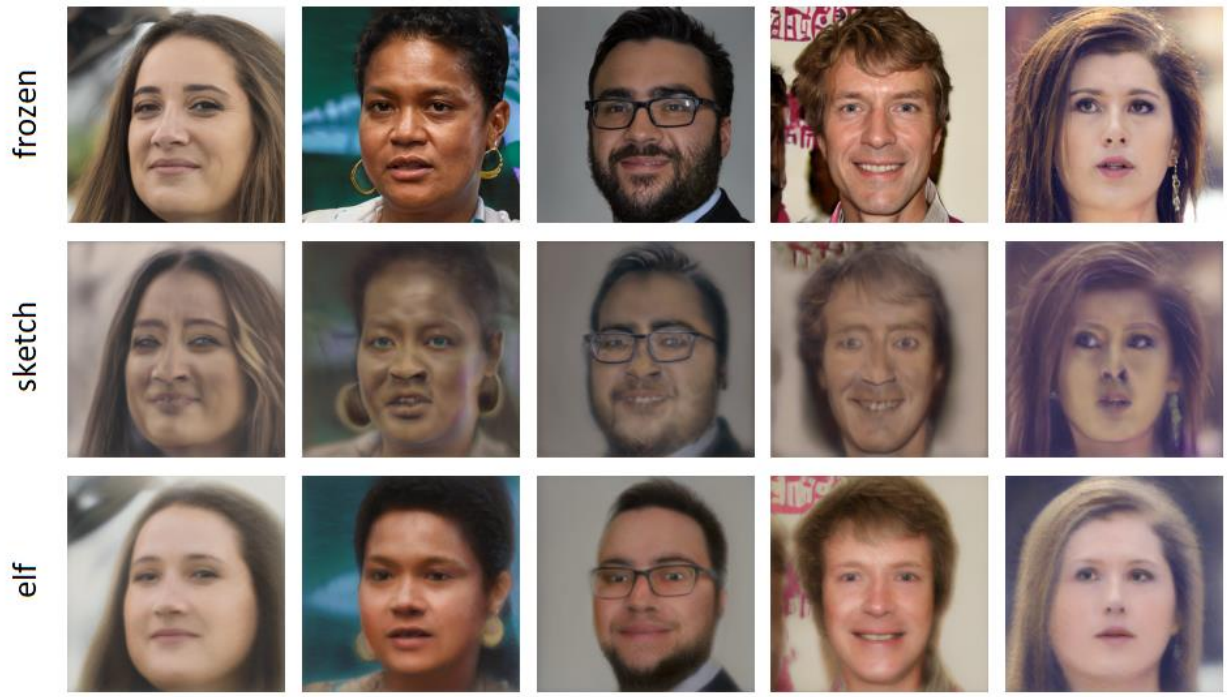

Рисунок 4. freeze_layers.

2. Адаптивный метод замораживания слоев (Оригинальный подход StyleGAN-NADA, Метод 2, `freeze_layers_adaptive`):

Подход: Использовался адаптивный метод замораживания слоев, как было предложено в оригинальной статье. Это означает, что на каждой итерации обучения выбираются только наиболее релевантные слои генератора для обновления, а остальные замораживаются. Этот метод пытается автоматически определить наиболее важные слои для модификации на основе их влияния на W-пространство (латентные коды).

Детали `freeze_layers_adaptive`**:**

- Создание W+ вектора: Генерируется случайный Z-вектор, преобразуется в W-вектор с помощью замороженного генератора и затем "размножается" до формы W+ (`latent_w_plus`), чтобы иметь отдельный `w` для каждого слоя генератора.
- Мини-оптимизация W-вектора: Выполняется несколько шагов градиентного спуска, чтобы найти модифицированный W-вектор (`latent_tensor`), который лучше соответствует целевому тексту в CLIP-пространстве. При этом сам генератор не обучается, оптимизируется только сам W-вектор.
- Оценка важности слоев: Измеряется, насколько сильно каждый компонент W+ вектора, соответствующий определенному слою генератора, изменился во время этой мини-оптимизации (путем вычисления абсолютной разницы `torch.abs(latent_tensor - latent_w_plus)`). Считается, что слои, требующие наибольшего изменения W-вектора для достижения цели, являются наиболее "важными" для текущей задачи стилизации.
- Выбор k наиболее важных слоев: Выбирается `k` слоев с наибольшими изменениями.
- Заморозка/разморозка генератора: Все параметры `model_train` по умолчанию сначала замораживаются. Затем размораживаются только параметры, принадлежащие выбранным `k` слоям.

Результат: Результат получился хорошим. Этот подход повышает стабильность обучения и эффективность, так как обучение концентрируется на наиболее влиятельных частях модели (Рис. 5).

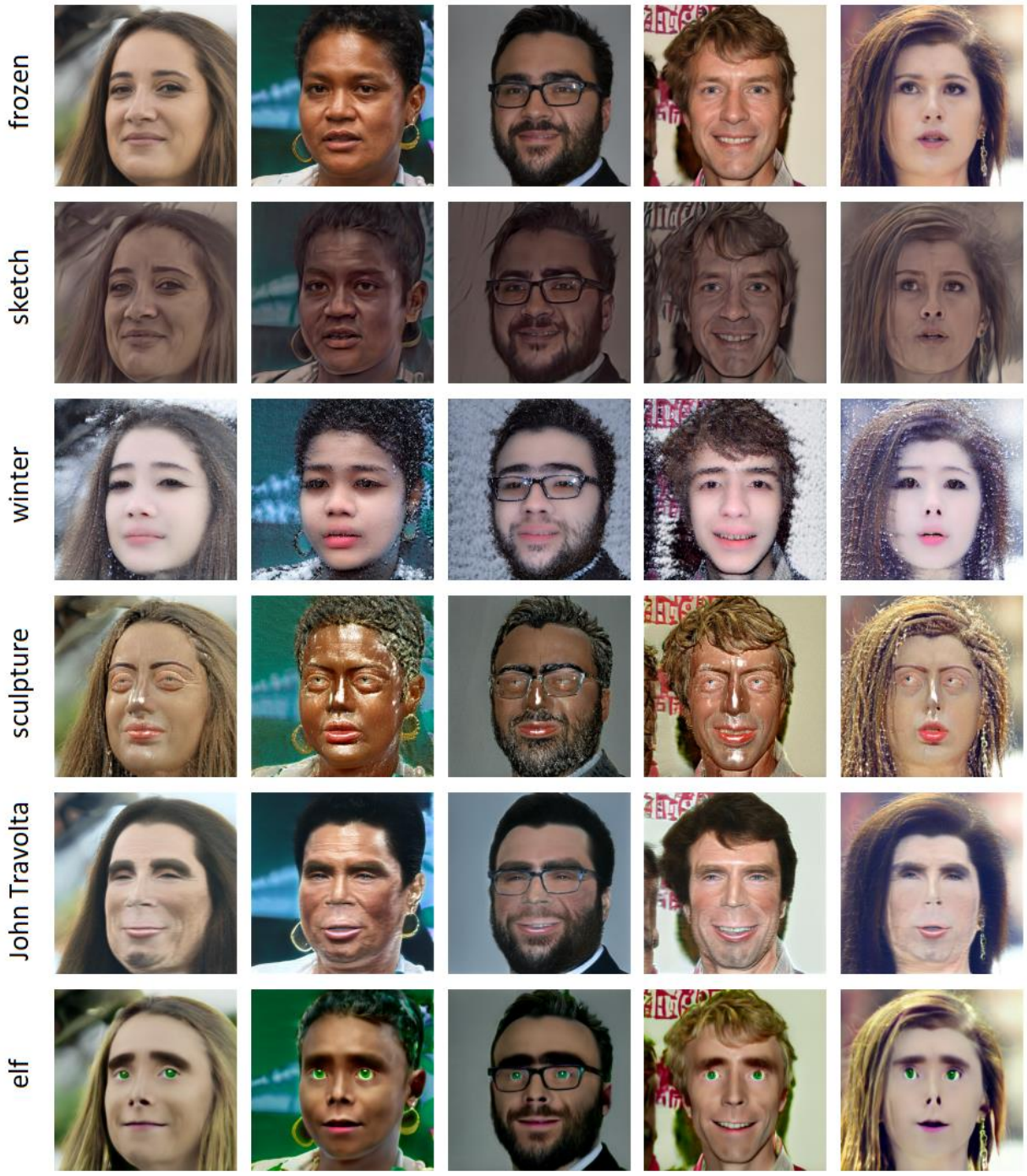

Рисунок 5. freeze_layers_adaptive. Адаптивная заморозка слоев.

3. Адаптивный метод с частичной разморозкой слоев (Эксперимент 3, `freeze_layers_adaptive_fine_grained`):

Подход: Опробован аналогичный адаптивный метод замораживания слоев (как в пункте 2), где также автоматически определяются наиболее важные слои для модификации на основе их влияния на W-пространство. Отличие заключается в том, что выбранные слои

размораживаются не полностью: параметры `weight` и `bias` (веса сверток и смещения) остаются замороженными, а параметры `modulation` и `noise` размораживаются.

Причина выбора `modulation` и `noise`**:** В StyleGAN `modulation` (адаптивная нормализация экземпляра) отвечает за применение стилей, контролируемых `w`-вектором, а `noise` добавляет стохастические детали. Обучение только этих параметров позволяет изменять стиль и детали, сохраняя при этом базовые "фильтры" (веса сверток), отвечающие за структурные особенности. Функция таким образом "гибко" размораживает только наиболее важные части модели, непосредственно отвечающие за применение стиля.

Результат: Результат получился неудовлетворительный. Это может указывать на то, что для достижения желаемых изменений домена, особенно если они затрагивают не только стиль/текстуру, но и более глобальные формы, необходимо размораживать и базовые веса сверточных слоев (Рис. 6).

Рисунок 6. freeze_layers_adaptive_fine_grained.

4. Адаптивный метод с выбором наиболее чувствительных параметров (Эксперимент 4, `freeze_layers_adaptive_fine_tune`**):**

Подход: Опробован способ более чувствительного адаптивного размораживания слоев, выделяя наиболее сильно меняющиеся параметры, а не слои.

Детали: На каждой эпохе функция вычисляет изменение параметров генератора при минимальной оптимизации (внутренний цикл с обучением *копии* тренировочного генератора), чтобы понять, какие конкретные веса действительно участвуют в направлении изображения к заданному тексту. Далее — `k` весов с наибольшими изменениями будут размерожены для дальнейшего обучения `model_train`.

Результат: Этот метод оказался слишком "точечным", не давая генератору достаточной гибкости для более обширных изменений домена (Рис. 7). К тому же результаты получаются не предсказуемые, так как недостаточно размораживать в каждой итерации только определенные параметры слоя. Это приводит к тому, что в процессе обучения `model_train` не делает необходимый сдвиг в визуальном пространстве для совпадения с направлением в текстовом пространстве.

Рисунок 7. freeze_layers_adaptive_fine_tune.

Подбор исходного текста (Source Text)

Поскольку CLIP обучался на обобщённых образах из интернета и может не интерпретировать визуальные изменения локально так, как мы ожидаем (например, из-за "смещения" в данных обучения), был опробован метод динамического обновления исходного текста (`source_text`) на каждой эпохе. Это обновление основывается на CLIP-описании изображения, сгенерированного замороженным генератором (Рис. 8).

Идея: CLIP "на лету" определяет, "что сейчас видит изображение", сгенерированное `model_frozen`, и именно это становится текущей отправной точкой (`source_text`).

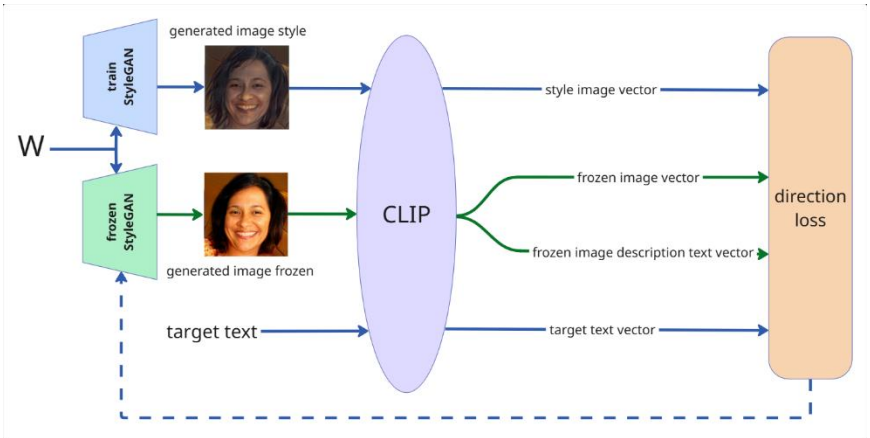

Рисунок 8. Подбор исходного текста с помощью CLIP.

Другими словами, мы хотим, чтобы направления между векторами изображения и текста не были слишком случайными или бессмысленными. Мы стремимся к тому, чтобы CLIP "понимал", куда мы хотим двигаться, и чтобы разница между `target_text` и `source_text` не была слишком мала, но при этом адекватно отражала текущее состояние `model_frozen`. Предполагается, что сдвиг, который модель делает в визуальном пространстве, будет лучше совпадать по направлению со сдвигом в текстовом пространстве, если `source_text` является динамически генерируемым описанием.

Для сравнения, этот метод был опробован с адаптивным методом замораживания слоев (`freeze_layers_adaptive`).

Результат: Полученные визуальные результаты не отличаются от использования адаптивным методом замораживания слоев (`freeze_layers_adaptive`) без динамического обновления исходного текста (`source_text`). Применение данного метода не добавило заметных преимуществ (Рис 9).

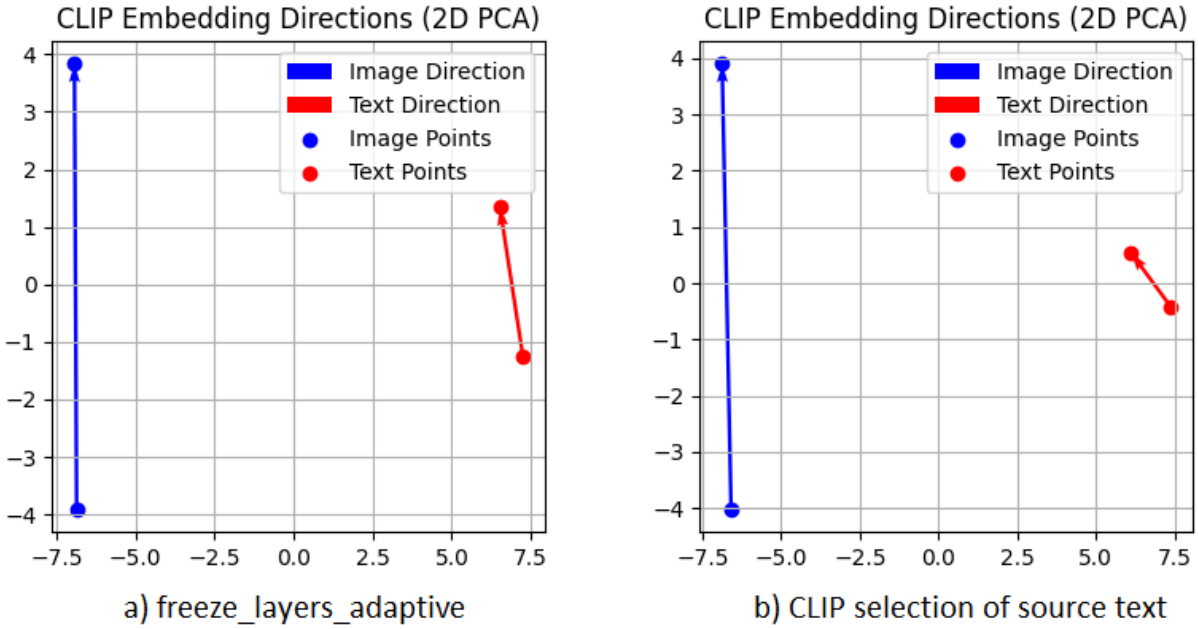

Рисунок 9. Визуализация направлений CLIP.

Выводы

Мы успешно воспроизвели и реимплементировали работу StyleGAN-NADA, научившись дообучать генератор для создания изображений из различных целевых доменов, основываясь исключительно на текстовых запросах.

Главные преимущества StyleGAN-NADA:

- **Отсутствие необходимости в обучающих изображениях** для целевого домена: достаточно лишь текстовой подсказки. Это значительно сокращает затраты на сбор и разметку данных.
- **Гибкость трансформаций:** Метод позволяет преобразовывать изображения в широкий спектр доменов, включая стилистические изменения, модификации формы и даже генерацию фантастических или абстрактных образов.
- **Высокая скорость обучения:** Благодаря файн-тюнингу относительно небольшого числа слоев (а не обучению с нуля) и эффективному использованию CLIP, процесс адаптации происходит значительно быстрее по сравнению с традиционным обучением GAN.
- **Сохранение свойств латентного пространства:** Метод сохраняет структуру латентного пространства StyleGAN, что позволяет выполнять дальнейшее редактирование сгенерированных изображений.
- **Сохранение качества генерации:** Мы убедились, что адаптивное замораживание/размораживание слоев генератора позволяет сохранить базовое качество генерации и структуру исходного изображения, поскольку неизмененные слои помогают поддерживать реалистичность.

Недостатки и ограничения:

- **Ограничения предварительно обученного генератора:** Модификации, которые метод может применять, в значительной степени ограничены областью предварительно обученного генератора StyleGAN2. Если целевой домен слишком сильно отличается от исходного, могут потребоваться более глубокие изменения или предварительное обучение на более разнообразном датасете.
- **Появление артефактов:** При очень сильных изменениях изображений или при адаптации к сложным/нереалистичным доменам могут появляться артефакты, такие как искажения или нереалистичные детали.
- **Разнообразие:** Для каждого стиля генерируемых изображений, необходимого отдельное обучение генератора.

Особенности использования CLIP:

Мы успешно использовали CLIP для редактирования изображений в направлении текстового запроса. Это гибкий и универсальный подход, который позволил нам обучить генератор только на текстовых запросах. Однако у CLIP есть несколько недостатков, которые накладывают ограничения:

1. **Ограниченность восприятия данными обучения:** CLIP ограничен данными, на которых он был обучен. Это может приводить к "галлюцинациям" или неточным интерпретациям для очень специфичных или редких визуальных концепций, не представленных в его обучающем наборе.
2. **Ограниченная точность семантического контроля:** CLIP не всегда точно понимает тонкие визуальные различия (например, "блестящие волосы" против "влажные волосы" могут быть интерпретированы схожим образом, если эти нюансы не были четко представлены в его обучающих данных).
3. **Обобщение смысла текста:** CLIP часто обобщает смысл текста, что может приводить к неожиданным или нежелательным результатам при редактировании, если пользователь ожидает очень специфическую трансформацию.
4. **Отсутствие обратной связи на пиксельном уровне:** В отличие от perceptual loss (например, VGG-loss), CLIP не сравнивает пиксели напрямую, а работает с глобальными эмбеддингами изображения и текста. Это означает, что результаты не всегда идеально совпадают с визуальными ожиданиями пользователя, даже если CLIP "думает", что результат верен семантически.

Опробованные конфигурации обучения:

Мы опробовали различные пользовательские конфигурации обучения. Наиболее удачными оказались:

- Автоматическая подборка коэффициентов функции потерь (λ_{CLIP} и λ_{L2}) с помощью отдельного оптимизатора.
- Адаптивное замораживание и размораживание слоев генератора (подход, описанный в пункте 2 "Адаптивный метод замораживания слоев").

Реализованное редактирование реальных изображений:

В сочетании с методом инверсии изображений (pSp/e4e (encoder4editing)), StyleGAN-NADA использован для редактирования реальных изображений (Рис. 10). Процесс включает инвертирование реального изображения в латентное пространство StyleGAN, а затем

применение адаптированного генератора для переноса этого изображения в новый домен, сохраняя при этом его идентичность.

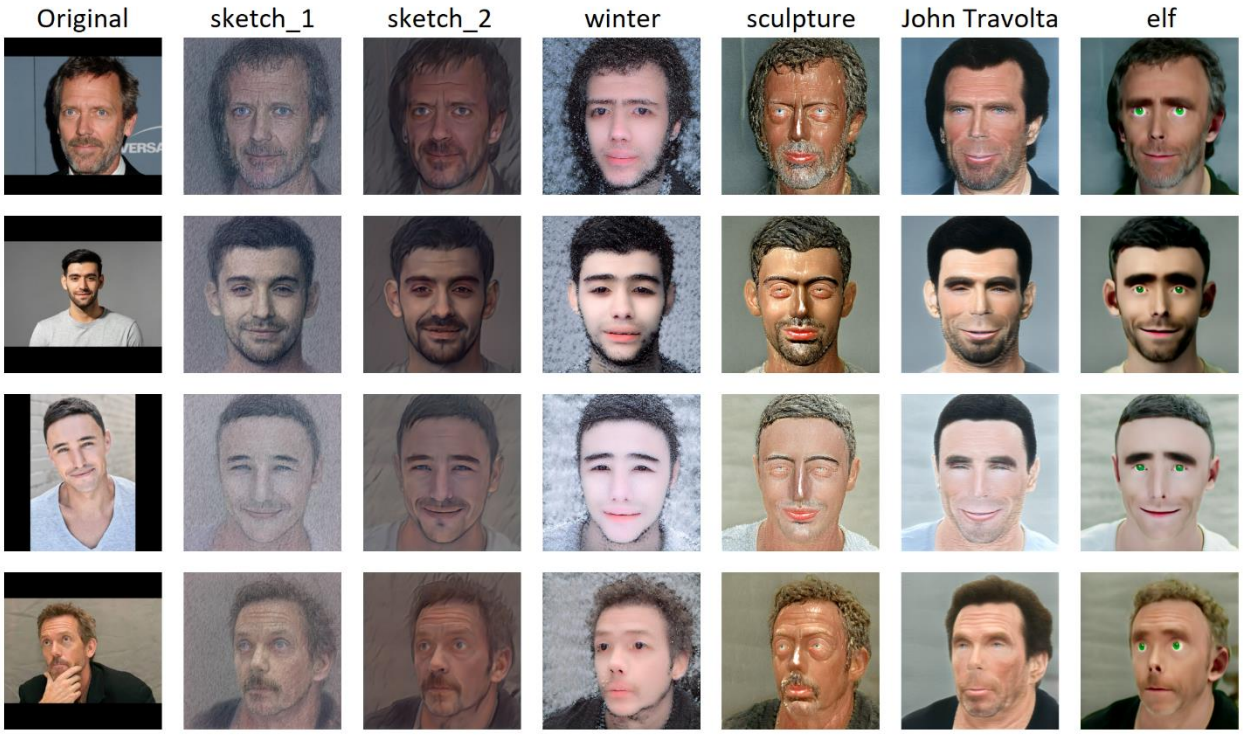

Рисунок 10. Редактирование реальных изображений.